Subject: Clarification Needed on Algorithms in OnionPIR Paper

Dear Mr. Mughees,

I hope this email finds you well. My name is Yue Chen, and I am a research assistant under Professor Ling Ren at the University of Illinois Urbana-Champaign. I am currently examining the OnionPIR protocol as detailed this paper (https://eprint.iacr.org/2021/1081) and have encountered some complexities with the QueryPack and QueryUnpack algorithms that I hope you could help clarify.

Firstly, I am uncertain about the input $\{b_i\}_{i=1}^{d}$ for QueryPack (algorithm 1). If each b_i is a vector of 0's and 1's, i.e., $b_i \in \mathbb{Z}_2^{N_i}$, the for-loop from line 5 to line 9, especially line 7, appears to have confusing notation. Additionaly, the third step in algorithm 3, states that "Client generates query vectors $\{b_j\}_{j=1}^{d}$ corresponding to $(i_1, \ldots, i_d)$ such that $b_j[i_j]$ is 1 and rest are 0", implying each b_j entry is a single bit.

In section 4.3 Query Compression, the text mentions "only pack l values, corresponding to first l rows of RGSW ciphertext". Could you elaborate on what these l values are? These values should have enough information for representing the first l rows of RGSW ciphertext, where each row is a polynomial ring element, i.e., each row $\in R \mod q$. q is the coefficient modulus for BFV ciphertext.

Another point of confusion arises in the first line of QueryUnpack (algorithm 2), where the expandRlwe algorithm from Onion Ring ORAM (https://eprint.iacr.org/2019/736) is used. The output of this expandRlwe is $c_i = \mathrm{RLWE}(n \cdot b_i), 0 \leq i < n$. Professor Ren and I are trying to understand how this factor of n is treated or removed in the QueryPack algorithm.

I appreciate your time in reviewing these details and any explanations you can provide would be immensely helpful.

Warm regards,

Yue Chen